**Scientific computing python libraries**

**Below are some of the libraries which have**

**Scientific programming toolkits**

* [PyEMD](https://pyemd.readthedocs.io/en/latest/intro.html) : PyEMD is a Python implementation of Empirical Mode Decomposition (EMD) and its variations. One of the most popular expansion is Ensemble Empirical Mode Decomposition (EEMD), which utilises an ensemble of noise-assisted executions.
* [Stable-Baselines3](https://stable-baselines3.readthedocs.io/en/master/): Stable Baselines3 (SB3) is a set of reliable implementations of reinforcement learning algorithms in PyTorch. It is the next major version of Stable Baselines.
* [Open AI gym](https://www.gymlibrary.dev/) : A toolkit for developing and comparing reinforcement learning algorithms.
* [Adaptive Experimentation platform (Facebook/Meta)](https://github.com/facebook/Ax) : Bayesian optimization and Multi-armed bandit library from Meta for performing general purpose optimization experiments.
* [Optuna](https://optuna.org/) : Hyperparameter optimization framework to automate hyperparameter search.
* [SciPy](https://www.scipy.org/) - Python modules for statistics, optimization, integration, linear algebra, etc. (Python, mostly BSD, [GitHub](https://github.com/scipy/scipy/))
* [NumPy](https://numpy.org/) - Fundamental package needed for scientific computing with Python. (Python, BSD, [GitHub](https://github.com/numpy/numpy))
* [PETSc](https://www.mcs.anl.gov/petsc/) - Parallel solution of scientific applications modeled by PDEs. (C, 2-clause BSD, [GitLab](https://gitlab.com/petsc/petsc))
* [DUNE Numerics](https://www.dune-project.org/) - Toolbox for solving PDEs with grid-based methods. (C++, GPL 2, [GitLab](https://gitlab.dune-project.org/core/))

**Data Visualization**

* [ParaView](https://www.paraview.org/) - Multi-platform data analysis and visualization application based on VTK. (C++, BSD, [GitLab](https://gitlab.kitware.com/paraview/paraview))
* [VTK](https://vtk.org/) - Process images and create 3D computer graphics. (C++, BSD, [GitLab](https://gitlab.kitware.com/vtk/vtk))
* [Mayavi](https://docs.enthought.com/mayavi/mayavi/) - 3D scientific data visualization and plotting in Python. (Python, BSD, [GitHub](https://github.com/enthought/mayavi))
* [Polyscope](https://polyscope.run/) - Viewer and user interface for 3D geometry processing. (C++, MIT, [GitHub](https://github.com/nmwsharp/polyscope))
* [PyVista](https://docs.pyvista.org/) - 3D plotting and mesh analysis through a streamlined interface for VTK. (Python, MIT, [GitHub](https://github.com/pyvista/pyvista))
* [vedo](https://vedo.embl.es/) - Library for scientific analysis and visualization of 3D objects based on VTK. (Python, MIT, [GitHub](https://github.com/marcomusy/vedo))
* [yt](https://yt-project.org/) - A toolkit for analysis and visualization of volumetric data. (Python, BSD, [GitHub](https://github.com/yt-project/yt))

**Finite Elements**

* [FEniCS](https://fenicsproject.org/) - Computing platform for solving PDEs in Python and C++. (C++/Python, LGPL 3, [Bitbucket](https://bitbucket.org/fenics-project/))
* [libMesh](https://libmesh.github.io/) - Framework for the numerical simulation of PDEs using unstructured discretizations. (C++, LGPL 2.1, [GitHub](https://github.com/libMesh/libmesh))
* [deal.II](https://dealii.org/) - Software library supporting the creation of finite element codes. (C++, LGPL 2.1, [GitHub](https://github.com/dealii/dealii))
* [Netgen/NGSolve](https://ngsolve.org/) - High performance multiphysics finite element software. (C++, LGPL 2.1, [GitHub](https://github.com/NGSolve/netgen))
* [Firedrake](https://www.firedrakeproject.org/) - Automated system for the solution of PDEs using the finite element method. (Python, LGPL 3, [GitHub](https://github.com/firedrakeproject/firedrake))
* [MOOSE](https://mooseframework.inl.gov/) - Multiphysics Object Oriented Simulation Environment. (C++, LGPL 2.1, [GitHub](https://github.com/idaholab/moose))
* [MFEM](https://mfem.org/) - Free, lightweight, scalable C++ library for finite element methods. (C++, LGPL 2.1, [GitHub](https://github.com/mfem/mfem))
* [SfePy](https://sfepy.org/) - Simple Finite Elements in Python. (Python, BSD, [GitHub](https://github.com/sfepy/sfepy))
* [FreeFEM](https://freefem.org/) - High level multiphysics-multimesh finite element language. (C++, LGPL, [GitHub](https://github.com/FreeFem))
* [libceed](https://libceed.readthedocs.io/en/latest/index.html) - Code for Efficient Extensible Discretizations. (C, 2-clause BSD, [GitHub](https://github.com/CEED/libCEED))

**Meshing**

* [Gmsh](http://gmsh.info/) - Three-dimensional finite element mesh generator with pre- and post-processing facilities. (C++, GPL, [GitLab](https://gitlab.onelab.info/gmsh/gmsh))
* [pygmsh](https://github.com/nschloe/pygmsh) - Python interface for Gmsh. (Python, GPL 3, GitHub)
* [MeshPy](https://mathema.tician.de/software/meshpy/) - Quality triangular and tetrahedral mesh generation. (Python, MIT, [GitHub](https://github.com/inducer/meshpy))
* [meshio](https://github.com/nschloe/meshio) - I/O for various mesh formats, file conversion. (Python, MIT, GitHub)
* [CGAL](https://www.cgal.org/) - Algorithms for computational geometry. (C++, mixed LGPL/GPL, [GitHub](https://github.com/CGAL/cgal))
* [pygalmesh](https://github.com/nschloe/pygalmesh) - Python interface for CGAL's 3D meshing capabilities. (Python, GPL 3, GitHub)
* [mshr](https://bitbucket.org/fenics-project/mshr/) - Mesh generation component of FEniCS. (Python, GPL 3, Bitbucket)
* [MOAB](https://press3.mcs.anl.gov/sigma/moab-library/) - Representing and evaluating mesh data. (C++, mostly LGPL 3, [Bitbucket](https://bitbucket.org/fathomteam/moab/))
* [NetCDF](https://www.unidata.ucar.edu/software/netcdf/) - Software libraries and data formats for array-oriented scientific data. (C/C++/Fortran/Java/Python, [custom open-source license](https://www.unidata.ucar.edu/software/netcdf/copyright.html), [GitHub](https://github.com/Unidata/netcdf-c/))
* [HDF5](https://support.hdfgroup.org/HDF5/) - Data model, library, and file format for storing and managing data. (C/Fortran, BSD)
* [XDMF](http://www.xdmf.org/index.php/Main_Page) - eXtensible Data Model and Format for data from High Performance Computing codes. (C++, [GitLab](https://gitlab.kitware.com/xdmf/xdmf))
* [TetGen](https://www.wias-berlin.de/software/index.jsp?id=TetGen) - Quality tetrahedral mesh generator and 3D Delaunay triangulator. (C++, AGPLv3)
* [Triangle](https://www.cs.cmu.edu/~quake/triangle.html) - Two-dimensional quality mesh generator and Delaunay triangulator. (C, *nonfree software*)
* [optimesh](https://github.com/nschloe/optimesh) - Triangular mesh smoothing. (Python, GPL 3, GitHub)
* [distmesh](http://persson.berkeley.edu/distmesh/) - Simple generator for unstructured triangular and tetrahedral meshes. (MATLAB, GPL 3)
* [QuadriFlow](https://stanford.edu/~jingweih/papers/quadriflow/) - A Scalable and Robust Method for Quadrangulation. (C++, BSD, [GitHub](https://github.com/hjwdzh/QuadriFlow))
* [trimesh](https://trimsh.org/) - Loading and using triangular meshes with an emphasis on watertight surfaces. (Python, MIT, [GitHub](https://github.com/mikedh/trimesh))
* [dmsh](https://github.com/nschloe/dmsh) - Simple generator for unstructured triangular meshes, inspired by distmesh. (Python, GPL 3, GitHub)
* [pmp-library](https://www.pmp-library.org/) - Polygon mesh processing library. (C++, MIT with Employer Disclaimer, [GitHub](https://github.com/pmp-library/pmp-library/))
* [Mmg](https://www.mmgtools.org/) - Robust, open-source & multidisciplinary software for remeshing. (C, LGPL 3, [GitHub](https://github.com/MmgTools/mmg))
* [meshplex](https://github.com/nschloe/meshplex) - Fast tools for simplex meshes. (Python, GPL 3, GitHub)
* [TetWild](https://cs.nyu.edu/~yixinhu/tetwild.pdf) - Robust Tetrahedral Meshing in the Wild. (C++, GPL 3, [GitHub](https://github.com/Yixin-Hu/TetWild))
* [TriWild](https://cims.nyu.edu/gcl/papers/2019-TriWild.pdf) - Robust Triangulation with Curve Constraints. (C++, MPL 2, [GitHub](https://github.com/wildmeshing/TriWild))
* [fTetWild](https://arxiv.org/abs/1908.03581) - Fast Tetrahedral Meshing in the Wild. (C++, MPL 2, [GitHub](https://github.com/wildmeshing/fTetWild))

**Sparse linear solvers**

* [SuperLU](https://portal.nersc.gov/project/sparse/superlu/) - Direct solution of large, sparse, nonsymmetric systems of linear equations. (C, mostly BSD, [GitHub](https://github.com/xiaoyeli/superlu))
* [KryPy](https://github.com/andrenarchy/krypy) - Krylov subspace methods for the solution of linear algebraic systems. (Python, MIT, GitHub)
* [PyAMG](https://pyamg.github.io/) - Algebraic Multigrid Solvers in Python. (Python, MIT, [GitHub](https://github.com/pyamg/pyamg))
* [hypre](https://computing.llnl.gov/projects/hypre-scalable-linear-solvers-multigrid-methods) - Library of high-performance preconditioners and solvers. (C, Apache 2.0/MIT, [GitHub](https://github.com/hypre-space/hypre))

**Other libraries and tools**

* [FFTW](http://www.fftw.org/) - Discrete Fourier transforms in one or more dimensions, of arbitrary input size, real and complex. (C, GPL2, [GitHub](https://github.com/FFTW/fftw3))
* [Qhull](http://www.qhull.org/) - Convex hull, Delaunay triangulation, Voronoi diagram, halfspace intersection about a point, etc. (C/C++, [custom open source license](http://www.qhull.org/COPYING.txt), [GitHub](https://github.com/qhull/qhull/))
* [GSL](https://www.gnu.org/software/gsl/) - Random number generators, special functions, and least-squares fitting etc. (C/C++, GPL 3, [Savannah](https://savannah.gnu.org/projects/gsl))
* [OpenFOAM](https://www.openfoam.com/) - Free, open source CFD (computational fluid dynamics) software. (C++, GPL 3, [GitHub](https://github.com/OpenFOAM/OpenFOAM-dev))
* [quadpy](https://github.com/nschloe/quadpy) - Numerical integration (quadrature, cubature) in Python. (Python, GPL 3, GitHub)
* [FiPy](https://www.ctcms.nist.gov/fipy/) - Finite-volume PDE solver. (Python, [custom open-source license](https://www.nist.gov/open/copyright-fair-use-and-licensing-statements-srd-data-software-and-technical-series-publications), [GitHub](https://github.com/usnistgov/fipy))
* [accupy](https://github.com/nschloe/accupy) - Accurate sums and dot products for Python. (Python, GPL 3, GitHub)
* [SLEPc](https://slepc.upv.es/) - Scalable Library for Eigenvalue Problem Computations. (C, 2-clause BSD, [GitLab](https://gitlab.com/slepc/slepc))
* [Chebfun](https://www.chebfun.org/) - Computing with functions to about 15-digit accuracy. (MATLAB, BSD, [GitHub](https://github.com/chebfun/chebfun))
* [pyMOR](https://pymor.org/) - Model Order Reduction with Python. (Python, 2-clause BSD, [GitHub](https://github.com/pymor/pymor/))
* [cvxpy](https://www.cvxpy.org/) - Modeling language for convex optimization problems. (Python, Apache 2.0, [GitHub](https://github.com/cvxgrp/cvxpy))
* [PyWavelets](https://pywavelets.readthedocs.io/en/latest/) - Wavelet transforms in Python. (Python, MIT, [GitHub](https://github.com/PyWavelets/pywt))
* [NFFT](https://www-user.tu-chemnitz.de/~potts/nfft/) - Nonequispaced fast Fourier transform. (C/MATLAB, GPL 2, [GitHub](https://github.com/NFFT/nfft))
* [preCICE](https://www.precice.org/) - Coupling library for partitioned multi-physics simulations (FSI, CHT, and more). (C++, LGPL 3, [GitHub](https://github.com/precice/))
* [orthopy](https://github.com/nschloe/orthopy) - Compute orthogonal polynomials efficiently. (Python, GPL 3, GitHub)

**Basic linear algebra**

* [BLAS](https://www.netlib.org/blas/) - Standard building blocks for performing basic vector and matrix operations. (Fortran, public domain, [GitHub](https://github.com/Reference-LAPACK/lapack/tree/master/BLAS))
* [OpenBLAS](https://www.openblas.net/) - Optimized BLAS library based on GotoBLAS2. (C and Assembly, BSD, [GitHub](https://github.com/xianyi/OpenBLAS))
* [BLIS](https://github.com/flame/blis) - High-performance BLAS-like dense linear algebra libraries. (C, BSD, GitHub)
* [LAPACK](https://www.netlib.org/lapack/) - Routines for solving systems of linear equations, linear least-squares, eigenvalue problems, etc. (Fortran, BSD, [GitHub](https://github.com/Reference-LAPACK/lapack))
* [Eigen](http://eigen.tuxfamily.org/index.php?title=Main_Page) - C++ template library for linear algebra. (C++, MPL 2, [GitLab](https://gitlab.com/libeigen/eigen))
* [Ginkgo](https://ginkgo-project.github.io/) - High-performance manycore linear algebra library, focus on sparse systems. (C++, BSD, [GitHub](https://github.com/ginkgo-project/ginkgo))
* [blaze](https://bitbucket.org/blaze-lib/blaze) - High-performance C++ math library for dense and sparse arithmetic. (C++, BSD, Bitbucket)

**Communities**

* [SciComp StackExchange](https://scicomp.stackexchange.com/) - Computational Science on the StackExchange network.
* [Wolfgang Bangerth's video class](https://www.math.colostate.edu/~bangerth/videos.html) - MATH 676: Finite element methods in scientific computing.
* [Nick Higham's blog](https://nhigham.com/) - Mostly on MATLAB, general computing advice.
* [Nick Trefethen's Video Lectures](https://people.maths.ox.ac.uk/trefethen/videos.html) - 36 video lectures on approximation theory/practice and scientific computing.
* [John D. Cook's blog](https://www.johndcook.com/blog/) - Feats of scientific computing.
* [Jack Dongarra's software list](https://www.netlib.org/utk/people/JackDongarra/la-sw.html) - List of freely available software for the solution of linear algebra problems.
* [NA Digest](https://www.netlib.org/na-digest-html/) - Collection of articles on topics related to numerical analysis and those who practice it.
* [Gabriel Peyré on Twitter](https://twitter.com/gabrielpeyre) - One tweet a day on computational mathematics.